IN THE CLAIMS:

1. (Cancelled)

2. (Cancelled)

3. (Cancelled)

4. (Cancelled)

5. (Cancelled)

6. (Cancelled)

7. (Cancelled)

8. (Cancelled)

9. (Cancelled)

10. (Cancelled)

11. (Cancelled)

12. (Cancelled)

13. (Original) A method for the direct call of a target function by means of a start function in a computer system comprising a processor with a memory management unit (MMU), wherein the computer system includes an operating system, wherein the start function is a component of a first task with a first memory context and the target function is in another memory context, the method comprising:

the first task executing the start function to perform[[ing]] a context switch from the first memory context into the other memory context;

executing the target function in the other memory context;

reversing said context switch to return to the first memory context after executing the target function.

14. (Original) The method according to Claim 13, wherein the CPU includes a MMU control register;

wherein the first task performing the context switch includes the first task writing the physical address of the memory context of the task containing the target function into the MMU control register.

15. (Original) The method according to Claim 13, further comprising

copying parts of the memory context of the first task as well as parts of the memory context of the second task containing the target function into a new memory context;

wherein the first task performing the context switch comprises the first task performing the context switch into the new memory context.

16. (New) A method for the direct call of a target function by means of a start function in a computer system comprising a processor with a memory management unit (MMU), wherein the computer system includes an operating system, wherein the start function is a component of a first task with a first memory context and the target function is in a second memory context, the method comprising:

building a third memory context;

copying parts of the memory context of the first task as well as parts of the memory context of the second memory context containing the target function into the third memory context;

the first task performing a context switch to the third memory context; and

executing the target function in the third memory context.

17. (New) The method of claim 16 further comprising:

reversing said context switch to return to the first memory context after executing the target function.

18. (New) The method of claim 16, further comprising:

locking the parts of a memory context of the first task as well as the parts of the memory context of the second task containing the target function;

wherein said locking occurs before said copying.

19. (New) The method of claim 18, further comprising:

unlocking the parts of a memory context of the first task as well as the parts of the memory context of the second task containing the target function;

wherein said unlocking occurs after said copying.

20. (New) The method of claim 16, further comprising:

the first task deactivating interrupt handling of the operating system.

21. (New) The method of claim 20,

wherein said deactivating includes changing a processor control register.

22. (New) The method of claim 16 wherein the target function does not execute a subroutine included in the operating system.

23. (New) The method of claim 16, further comprising:

blocking a second executing of the target function during said executing the target function.

24. (New) A system for performing a direct call of a target function, wherein the system is operable to execute on a processor with a memory management unit (MMU) in a computer operated by an operating system, the system comprising:

a start function operable to be executed, wherein the start function is a component of a first task with a first memory context;

the target function operable to be executed, wherein the target function is in a second memory context, wherein the first task performs a context switch from the first memory context into the second memory context, and wherein the context switch is reversed after the execution of the target function.